

Git Bash

- Faça download no link: <https://git-scm.com/downloads>

The image consists of two screenshots of the Git website. The top screenshot shows the 'Downloads' page. The 'Windows' link is highlighted with a red box. The bottom screenshot shows the 'Download for Windows' page. The '64-bit Git for Windows Setup' link is highlighted with a red box.

Git --fast-version-control

Search entire site...

About

Documentation

Downloads

GUI Clients

Logos

Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

Downloads

Older releases are available and the Git source repository is on GitHub.

Latest source Release
2.37.1
Release Notes (2022-07-04)
Download for Windows

GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

Git via Git

If you already have Git installed, you can get the latest development version via Git itself:

```
git clone https://github.com/git/git
```

You can also always browse the current contents of the git repository using the [web interface](#).

Git --distributed-even-if-your-workflow-isnt

Search entire site...

About

Documentation

Downloads

GUI Clients

Logos

Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

Download for Windows

Click here to download the latest (2.37.1) 64-bit version of Git for Windows. This is the most recent maintained build. It was released 29 days ago, on 2022-07-12.

Other Git for Windows downloads

Standalone Installer

32-bit Git for Windows Setup.

64-bit Git for Windows Setup.

Portable ("thumbdrive edition")

32-bit Git for Windows Portable.

64-bit Git for Windows Portable.

Using winget tool

Install winget tool if you don't already have it, then type this command in command prompt or Powershell.

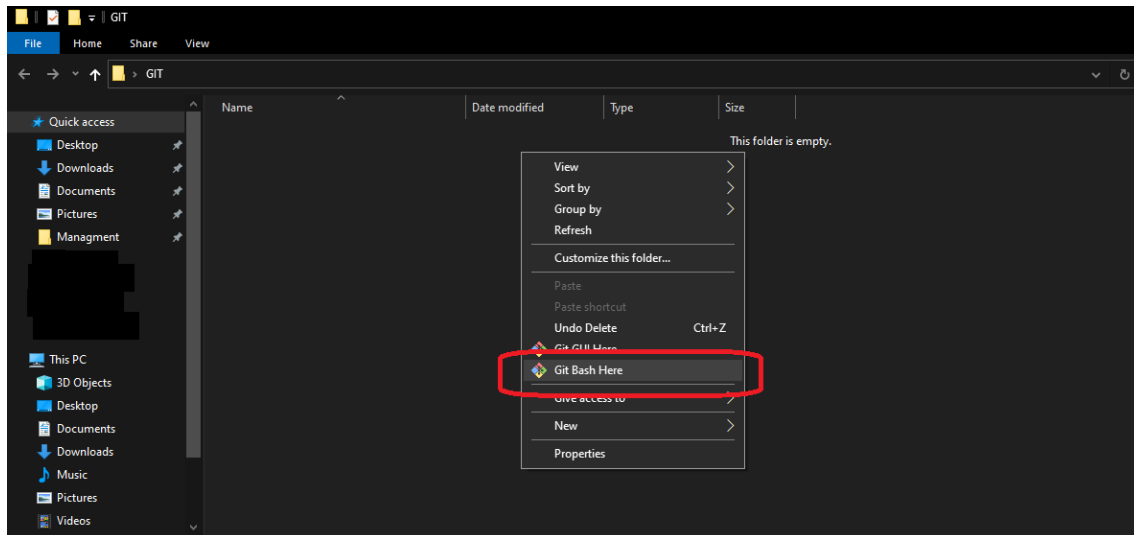
```
winget install --id Git.Git -e --source winget
```

The current source code release is version 2.37.1. If you want the newer version, you can build it from the source code.

Now What?

Now that you have downloaded Git, it's time to start using it.

- Instale o GIT.
- Na pasta em que você deseja realizar o download do repositório criado, abra o Git Bash .



Vamos configurar suas credenciais:

Use comando:

```
git config --global user.email "email_github@gmail.com"
```

Em seguida use o comando (O seu user name é visível na sua conta do GitHub):

```
git config --global user.name "name_do_github"
```

```
MINGW64/c:/Users/Mari/Desktop/GIT/nome-do-trabalho
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/nome-do-trabalho (main)
$ git config --global user.email "email_github@gmail.com"
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/nome-do-trabalho (main)
$ git config --global user.name "name_do_github"
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/nome-do-trabalho (main)
$
```

Vamos abrir um parêntese aqui:

Os seguintes comandos são importantes:

- `ls` : quando você pretender lista as pasta ou arquivos.
- `cd nome_pasta/` : quando você pretender entrar em uma pasta específica.
- `cd ..` : volta uma pasta.

Exemplo:

```

MINGW64/c/Users/Mari/Desktop/GIT
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT
$ ls
nome-do-trabalho/ pasta1/ pasta2/
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT
$ cd pasta1
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/pasta1
$ ls
pasta3/
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/pasta1
$ cd pasta3/
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/pasta1/pasta3
$ ls
texto.txt
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/pasta1/pasta3
$ cd ..
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/pasta1
$ cd ..
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT
$ ls
nome-do-trabalho/ pasta1/ pasta2/
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT
$

```

Estamos dentro da pasta GIT.

Primeiro, listamos o que há dentro da pasta GIT.

Temos a pasta nome-do-trabalho, pasta1 e pasta2.

Depois a pasta1 foi acessada.

Dentro da pasta1 listamos o que há dentro dela, encontramos a pasta3.

A pasta3 foi acessada.

Dentro da pasta3 listamos o que há dentro dela, encontramos um arquivo de texto.

Voltamos da pasta3 para a pasta1

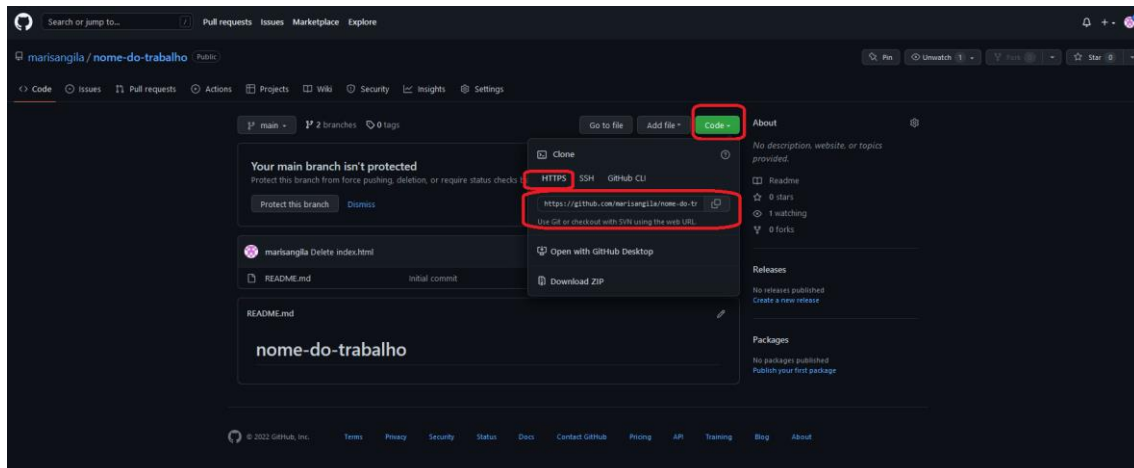
Voltamos da pasta1 para a pasta inicial (GIT)

Listamos novamente, para visualizar o que há na pasta GIT

Seguindo:

Agora vamos **clonar** seu repositório no GitHub para o computador:

- Copie a URL do seu repositório



- Agora novamente no Git Bash use o comando:

```
git clone "link_que_voce_copiou"
```

```
MINGW64: c:/Users/Mari/Desktop/GIT
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT
$ git clone https://github.com/marisangila/nome-do-trabalho.git
Cloning into 'nome-do-trabalho'...
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 12 (delta 1), reused 6 (delta 0), pack-reused 0
Receiving objects: 100% (12/12), done.
Resolving deltas: 100% (1/1), done.

Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT
$
```

- Entre na pasta:

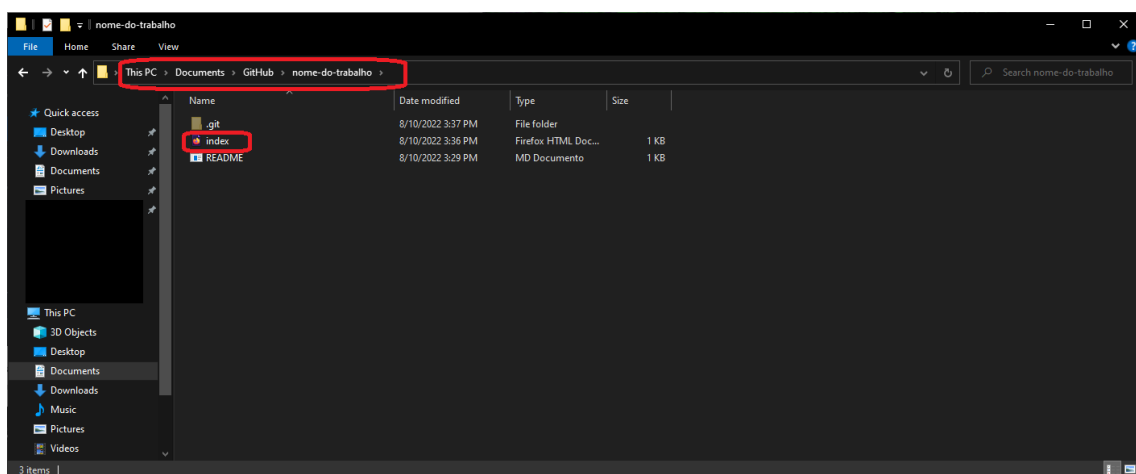
```
MINGW64: c:/Users/Mari/Desktop/GIT/nome-do-trabalho
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT
$ ls
nome-do-trabalho/

Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT
$ cd nome-do-trabalho/

Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/nome-do-trabalho (main)
$
```

Vamos criar um arquivo de teste

- Crie um arquivo qualquer dentro da pasta do seu repositório:



- Agora use o comando:

```
git status
```

O comando **git status** deve ser usado sempre que você precisar chegar algo (Exemplo: quais arquivos foram editados).

O novo arquivo aparecerá em como **untracked files** (vermelho):

```
MINGW64/c/Users/Mari/Desktop/GIT/nome-do-trabalho
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/nome-do-trabalho (tela_login)
$ git status
On branch tela_login
Untracked files:
  (use "git add <file>..." to include in what will be committed)
  index.html
nothing added to commit but untracked files present (use "git add" to track)
```

- Adicione o arquivo a **área de staging** usando o comando:

```
git add "nome_arquivo.html"
```

Se você usar o comando **git status** novamente verá que o arquivo como **new file** (verde):

```
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/nome-do-trabalho (tela_login)
$ git add index.html
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/nome-do-trabalho (tela_login)
$ git status
On branch tela_login
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
  new file:   index.html
```

Agora é necessário realizar um **commit** usando o comando:

```
git commit -m "meu primeiro commit"
```

No **commit** você define quais arquivos ou alterações serão enviadas ao GitHub, além disso, você pode definir uma mensagem.

(Lembre-se usar aqui uma mensagem sinalizando qual alteração você fez, assim fica fácil mapear alterações futuramente).

```
MINGW64/c/Users/Mari/Desktop/GIT/nome-do-trabalho
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/nome-do-trabalho (main)
$ git commit -m "meu primeiro commit"
[main fbf5fcd] meu primeiro commit
1 file changed, 3 insertions(+)
create mode 100644 index.html
```

Antes de enviar suas alterações ao GitHub é necessário garantir que os arquivos na sua máquina estão com as últimas alterações existentes no repositório do GitHub (Outro integrante da equipe pode ter realizado alguma alteração).

- Para isso use o comando:

```
git pull
```

```
MINGW64/c/Users/Mari/Desktop/GIT/nome-do-trabalho
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/nome-do-trabalho (main)
$ git pull
Already up to date.
```

Caso tenha atualizações, o git irá realizar download, caso contrário aparecerá a mensagem **“Already up to date”**

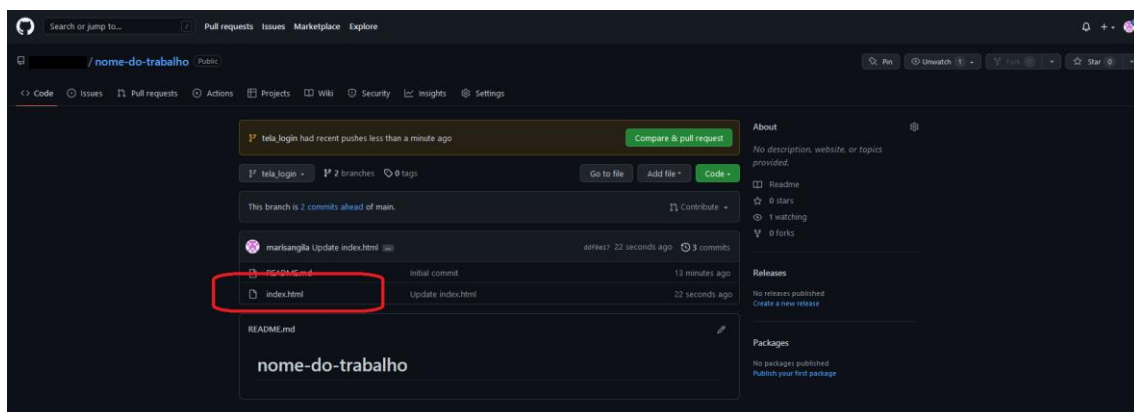
Finalmente, é possível enviar suas alterações para o repositório no GitHub, assim os outros integrantes da equipe poderão ver suas alterações.

- Para isso use o comando:

```
git push
```

```
MINGW64/c:/Users/Mari/Desktop/GIT/nome-do-trabalho
Mari@DESKTOP-L09KE5R MINGW64 ~/Desktop/GIT/nome-do-trabalho (tela_login)
$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 295 bytes | 295.00 KiB/s, done.
Total 3 (delta 0), reused 2 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'tela_login' on GitHub by visiting:
remote:   https://github.com/marisangila/nome-do-trabalho/pull/new/tela_login
remote:
To https://github.com/marisangila/nome-do-trabalho.git
 * [new branch]      tela_login -> tela_login
```

Agora, suas alterações apareceram no GitHub!



Agora os outros integrantes podem ver suas alterações e transferir essas alterações para o computador usando **GIT PULL**!